

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ
по дисциплине «Нереляционные базы данных»

Тема: каталог морских путешествий из старых корабельных журналов

| | | |
|------------------|-------|-----------------|
| Студент гр. 7304 | _____ | Есиков О.И. |
| Студент гр. 7304 | _____ | Моторин Е.В. |
| Студент гр. 7304 | _____ | Пэтайчук Н.Г. |
| Преподаватель | _____ | Заславский М.М. |

Санкт-Петербург
2020

ЗАДАНИЕ

НА ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

Студенты

Есиков О.И.

Моторин Е.В.

Пэтайчук Н.Г.

группа 7304

Тема работы: каталог морских путешествий из старых корабельных журналов.

Исходные данные:

Создание приложения, в функциональность которого входят добавление путешествий, оценка средних параметров путешествий, фильтрация, сопоставление данных.

Содержание пояснительной записки:

«Содержание»

«Введение»

«Сценарий использования»

«Модель данных»

«Разработка приложения»

«Вывод»

«Приложение»

Предполагаемый объем пояснительной записки:

Не менее 25 страниц.

Дата выдачи задания: 18.09.2020

Дата сдачи реферата:

Дата защиты реферата:

Студент гр. 7304

Есиков О.И.

Студент гр. 7304

Моторин Е.В.

Студент гр. 7304

Пэтайчук Н.Г.

Преподаватель

Заславский М.М.

АННОТАЦИЯ

В рамках данного курса предполагалось какое-либо приложение в команде на одну из поставленных тем. Была выбрана тема создания приложения каталог морских путешествий из старых корабельных журналов.

Найти исходный код и дополнительную информацию можно по ссылке:
<https://github.com/moevm/nosql2h20-sea-trips>.

SUMMARY

Within the framework of this course, it was assumed that an application in a team was one of the set topics. The theme was chosen to create a sea travel app from old magazines.

You can find the source code and additional information here:
<https://github.com/moevm/nosql2h20-sea-trips>.

СОДЕРЖАНИЕ

| | |
|--|----|
| 1. ВВЕДЕНИЕ | 6 |
| 2. КАЧЕСТВЕННЫЕ ТРЕБОВАНИЯ К РЕШЕНИЮ | 7 |
| 3. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ | 8 |
| 4. МОДЕЛЬ ДАННЫХ | 22 |
| 5. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ..... | 29 |
| 6. ВЫВОДЫ..... | 31 |
| 7. ПРИЛОЖЕНИЯ | 32 |
| 8. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ | 33 |

I. ВВЕДЕНИЕ

Целью работы является создание приложения, в функциональность которого входят добавление путешествий, оценка средних параметров путешествий, фильтрация, сопоставление данных. Выбранный нами стек технологий включает в себя Node.js, Vue.js[1], CSS, Express.js[3], Mongo[2].

II. КАЧЕСТВЕННЫЕ ТРЕБОВАНИЯ К РЕШЕНИЮ

Требуется разработать user-friendly приложение, в функциональность которого будут входить: страница со списком морских путешествий, содержащая записи за все время, статистика путешествий, фильтрация и сортировка путешествий, добавление путешествий, импорт и экспорт данных БД. В качестве системы управления базами данных использовать MongoDB [2].

III. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

3.1 Макеты UI

1. Просмотр списка морских путешествий.

The screenshot shows a web browser window with the address bar displaying 'http://localhost:3000'. The page title is 'Sea Voyages Journal'. Below the title, there are four buttons: 'ADD NEW RECORD', 'RECORDS FILTER AND SORT', 'STATISTICS', and 'BACK-UP USAGE'. Below these buttons is a table with the following data:

| Ship Name | Commander | Departure | Destination | Start Date | End Date | Distance, km | |
|-----------|--------------|-----------|---------------|------------|------------|--------------|--------|
| LEWIS | D.HUTCHINSON | SALEM | ZANZIBAR | 1854/01/01 | 1854/02/01 | 465 | Delete |
| LEWIS | D.HUTCHINSON | ZANZIBAR | MUSCAT | 1854/03/12 | 1854/04/12 | 567 | Delete |
| COVINGA | CHAS.N.BATES | NEW YORK | SAN FRANCISCO | 1853/12/04 | 1854/01/04 | 2134 | Delete |

Below the table is a pagination control with buttons for 1, 2, ..., 10, 20, and a greater-than sign (>).

Рисунок 3.1. — Просмотр списка морских путешествий.

2. Добавление новой записи.

The screenshot shows a form titled 'ADD NEW RECORD' with a close button (X) in the top right corner. The form contains the following fields:

- Ship name: Black Pearl
- Commander: Cpt. Jack Sparrow
- Departure: Tartuga
- Destination: London
- Start Date: 1855/01/13 (with a calendar icon)
- End Date: 1855/02/20 (with a calendar icon)
- Distance, km: 2000

At the bottom right of the form is a 'SUBMIT' button.

Рисунок 3.2. — Добавление новой записи.

3. Добавление фильтров и параметров сортировки записей.

FILTER AND SORTING

SORT BY:

DISTANCE

▼

REVERSE

▼

ADD NEW SORTING FIELD

FILTER BY:

Ship name:

Commander:

Cpt. Jack Sparrow

Departure:

Destination:

Start Date:

1855/01/13

...

End Date:

1855/02/20

...

Distance, km:

PERFORM

Рисунок 3.3. — Добавление фильтров и параметров сортировки записей.

4. Просмотр путешествия.

Moqzilla

← → ↻

http://localhost:3000/recrod/3

Sea Voyages Journal: Record №3

Ship name:

COVINGA

Commander:

CHAS.N.BATES

NEW YORK
1853/12/04

→

2134 km

→

SAN RANCISCO
1804/01/04

RETURN TO MAIN PAGE

Рисунок 3.4. — Просмотр путешествия.

5. Просмотр статистики путешествий.

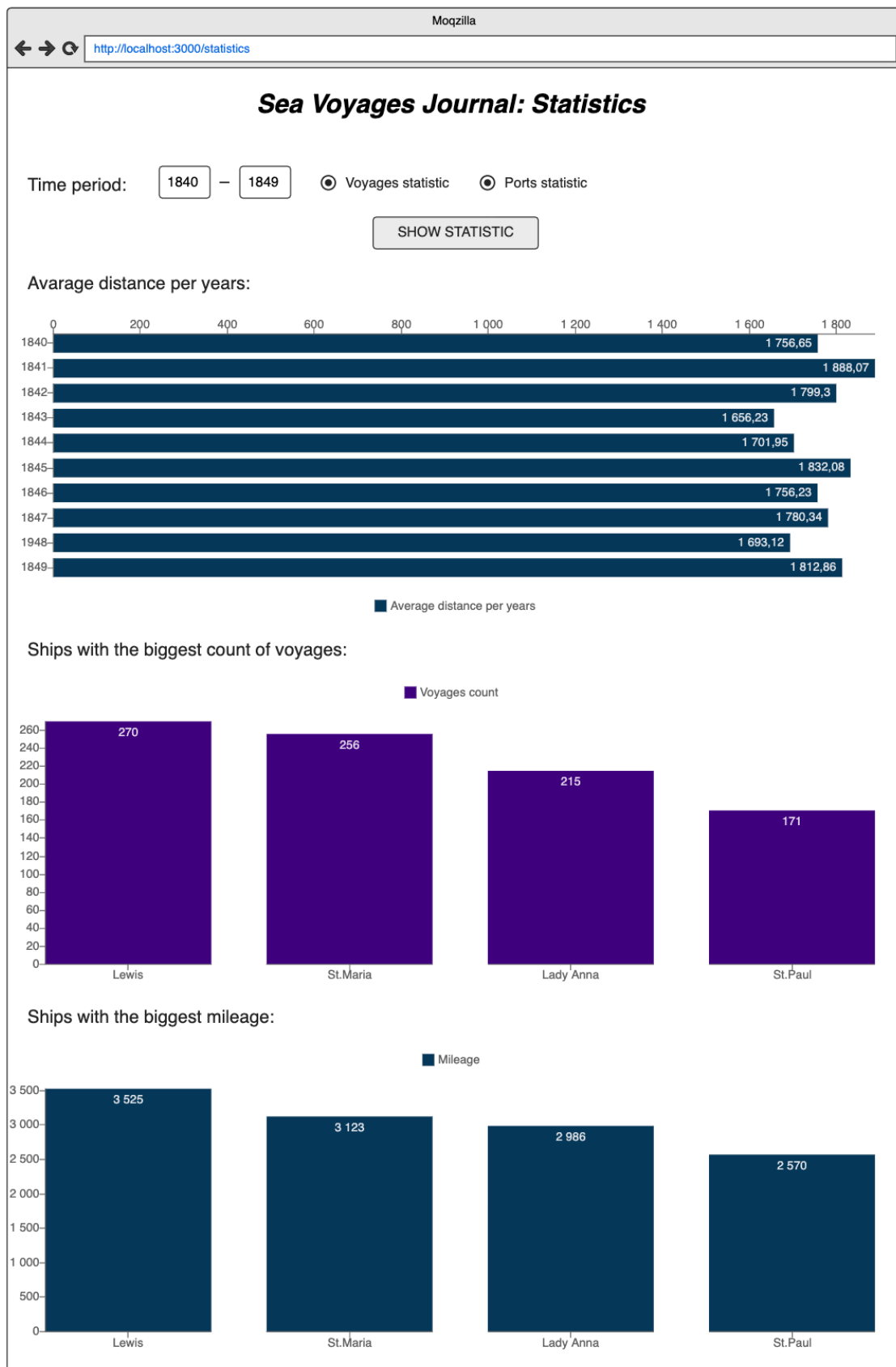


Рисунок 3.5 — Просмотр статистики путешествий.

6. Просмотр статистики портов.

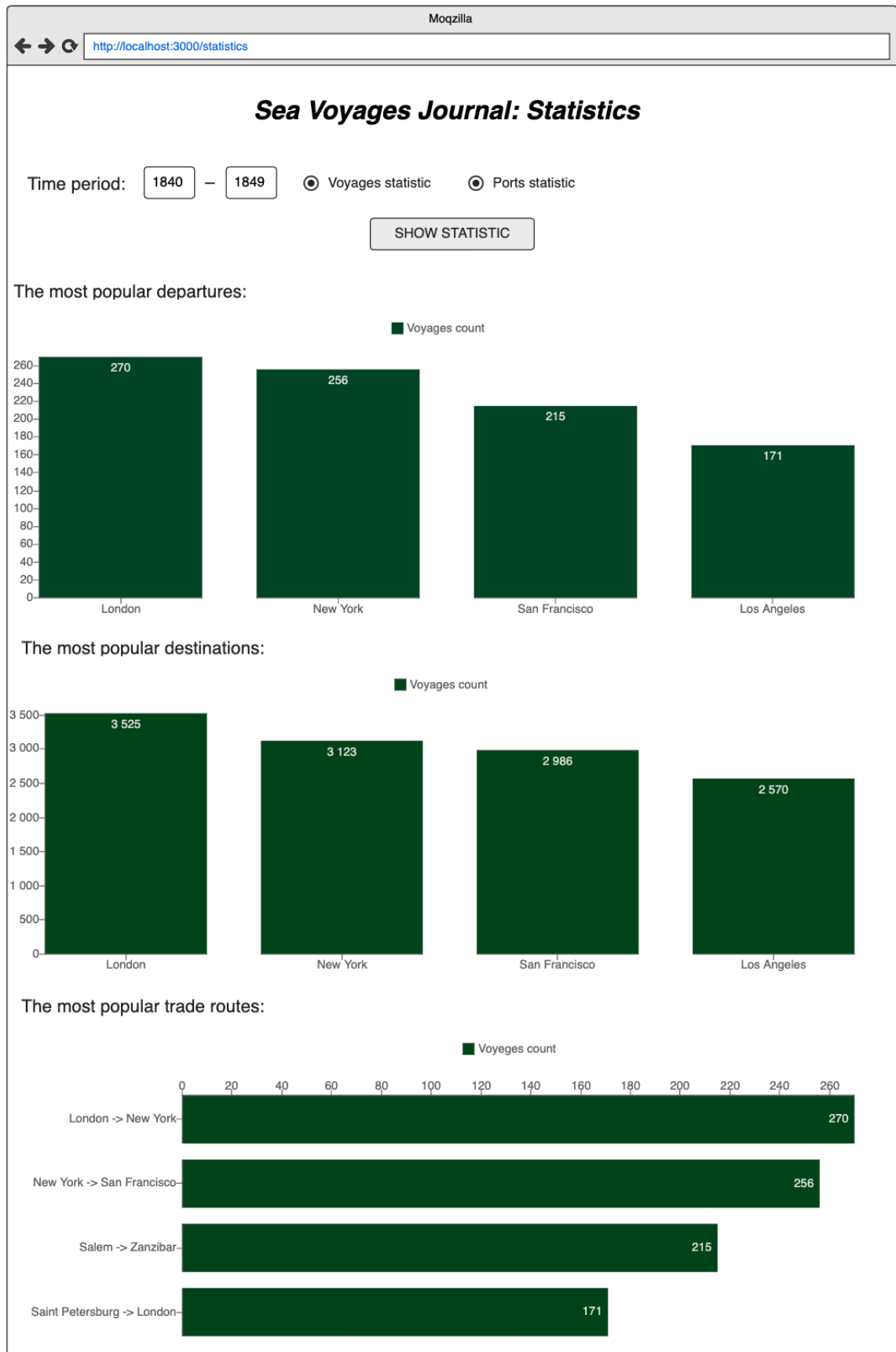


Рисунок 3.6 — Просмотр статистики портов.

7. Подробная информация по гистограмме путешествий.

| Ship Records | | | | | | |  |
|---------------------|--------------|-----------|---------------|------------|------------|--------------|---|
| Ship Name | Commander | Departure | Destination | Start Date | End Date | Distance, km | |
| LEWIS | D.HUTCHINSON | SALEM | ZANZIBAR | 1854/01/01 | 1854/02/01 | 465 | |
| LEWIS | D.HUTCHINSON | ZANZIBAR | MUSCAT | 1854/03/12 | 1854/04/12 | 567 | |
| LEWIS | CHAS.N.BATES | NEW YORK | SAN FRANCISCO | 1853/12/04 | 1854/01/04 | 2134 | |

Рисунок 3.7 — Подробная информация по гистограмме путешествий.

8. Подробная информация по гистограмме портов.



| Port Records | | | | | | |  |
|---------------------|--------------|-----------|---------------|------------|------------|--------------|---|
| Ship Name | Commander | Departure | Destination | Start Date | End Date | Distance, km | |
| LEWIS | D.HUTCHINSON | NEW YORK | ZANZIBAR | 1854/01/01 | 1854/02/01 | 465 | |
| LEWIS | D.HUTCHINSON | NEW YORK | MUSCAT | 1854/03/12 | 1854/04/12 | 567 | |
| COVINGA | CHAS.N.BATES | NEW YORK | SAN FRANCISCO | 1853/12/04 | 1854/01/04 | 2134 | |

Рисунок 3.8 — Подробная информация по гистограмме портов.

9. Импорт/экспорт БД.

BACK-UPS 

File name:

Рисунок 3.9 — Импорт/экспорт БД.

10. Удаление путешествия.

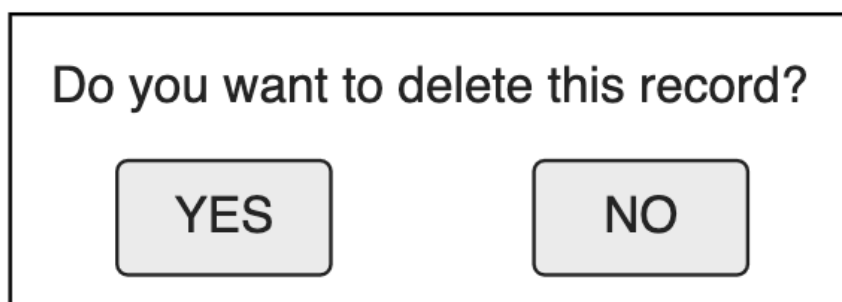


Рисунок 3.9 — Импорт/экспорт БД.

3.2 Сценарии использования.

Сценарий – «Просмотр журнала морских путешествий».

Действующее лицо: Пользователь

Основной сценарий:

1. Пользователь открывает главную страницу приложения.
2. Пользователь просматривает журнал морских путешествий в таблице.
3. Пользователь переходит на следующую страницу в таблице с помощью кнопок постраничной пагинации под таблицей.

Альтернативный сценарий:

1. Пользователь получает журнал морских путешествий через API.

Сценарий – «Просмотр конкретного путешествия».

Действующее лицо: Пользователь

Основной сценарий:

1. Пользователь открывает главную страницу приложения.

2. Пользователь просматривает журнал морских путешествий в таблице.

3. Пользователь нажимает на строку с путешествием в таблице.

Результат:

1. Осуществляется переход на страницу с информацией о путешествии.

Альтернативный сценарий:

1. Пользователь получает журнал морских путешествий через API.

Сценарий – «Удаление записи из журнала».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения.

Основной сценарий:

1. Пользователь находит в таблице нужную ему запись.

2. Пользователь нажимает на кнопку «Delete» в строке нужной ему записи.

3. Открывается модальное окно с подтверждением удаления записи.

4. Пользователь нажимает «Yes».

Альтернативный сценарий:

1. Пользователь в модальном окне нажимает кнопку «No» и отказывается от удаления.

2. Пользователь удаляет запись через API.

Результат:

1. Из таблицы пропала запись, для которой была нажата кнопка «Delete».

Сценарий – «Добавление новой записи в журнал».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения.

Основной сценарий:

1. В открывшемся модальном окне пользователь вводит данные:
 - a. В поле «Ship Name» название судна.
 - b. В поле «Commander» имя капитана судна.
 - c. В поле «Departure» порт отправления.
 - d. В поле «Destination» порт назначения.
 - e. В поле «Start Date» дату отплытия.
 - f. В поле «End Date» дату прибытия.
 - g. В поле «Distance, km» протяжённость маршрута.
2. Пользователь нажимает кнопку «SUBMIT».

Альтернативный сценарий:

1. Пользователь закрывает модальное окно без нажатия на кнопку «SUBMIT».
2. Пользователь вводит данные не во все поля модального окна.
3. Пользователь добавляет маршрут через API.

Результат:

1. В таблице появилась новая запись.

Сценарий – «Установка фильтра и/или сортировки».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения.

Основной сценарий:

1. Пользователь нажимает на кнопку «RECORDS FILTER AND SORT»
2. В открывшемся модальном окне в разделе «SORT BY» пользователь выбирает в двух выпадающих списках поле, по которому произвести сортировку, и порядок сортировки.
3. Если пользователю необходимо отсортировать ещё по одному полю, то он нажимает на кнопку «ADD NEW SORTING FIELD» и переходит на шаг 2.
4. В открывшемся модальном окне в разделе «FILTER BY» пользователь указывает данные в тех полях, по которым хочет произвести фильтрацию:
 - a. В поле «Ship Name» название судна.
 - b. В поле «Commander» имя капитана судна.
 - c. В поле «Departure» порт отправления.
 - d. В поле «Destination» порт назначения.
 - e. В поле «Start Date» дату отплытия.
 - f. В поле «End Date» дату прибытия.
 - g. В поле «Distance, km» протяжённость маршрута.
5. Пользователь нажимает кнопку «PERFORM».

Альтернативный сценарий:

1. Пользователь закрывает модальное окно без нажатия на кнопку «PERFORM».
2. Пользователь выполняет фильтрацию и/или сортировку через API.

Результат:

1. В таблице произвелась фильтрация и/или сортировка по указанным значениями.

Сценарий – «Загрузка бэкапа данных».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения.

Основной сценарий:

1. Пользователь нажимает на кнопку «BACK-UP USAGE».
2. В открывшемся модальном окне с помощью кнопки «REVIEW» пользователь выбирает в файловой системе нужный файл с данными.
3. Пользователь нажимает на кнопку «IMPORT DB FROM FILE»

Альтернативный сценарий:

1. Пользователь закрывает модальное окно без нажатия на кнопку «IMPORT DB FROM FILE».
2. Пользователь нажимает на кнопку «EXPORT DB TO FILE».
3. Пользователь загружает пустой файл.
4. Пользователь загружает файл, данные в котором не содержат названия полей из коллекции в бд.

5. Пользователь выполняет загрузку бэкапа через API.

Результат:

1. Журнал морских путешествий теперь содержит данные из загруженного файла.

Сценарий – «Выгрузка бэкапа данных».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения.

Основной сценарий:

1. Пользователь нажимает на кнопку «BACK-UP USAGE».
2. В открывшемся модальном окне с помощью кнопки «REVIEW» пользователь выбирает в файловой системе файл, куда будет произведена выгрузка.
3. Пользователь нажимает на кнопку «EXPORT DB TO FILE»

Альтернативный сценарий:

1. Пользователь закрывает модальное окно без нажатия на кнопку «EXPORT DB TO FILE».
2. Пользователь нажимает на кнопку «IMPORT DB FROM FILE».
3. Пользователь не имеет достаточно свободной памяти для выгрузки файла на своё устройство.
4. Пользователь не дожидается окончания выгрузки файла.
5. Пользователь выполняет выгрузку бэкапа через API.

Результат:

1. Пользователь имеет файл с выгруженными данными из бд.

Сценарий – «Просмотр статистики по кораблям».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения и переходит на страницу «statistics» с помощью кнопки «STATISTICS».
2. Пользователь находится на странице «statistics».

Основной сценарий:

1. Пользователь вводит промежуток в годах в два поля, которые имеют подпись «Time period».
 2. В устанавливает делает активным флажок «Voyages statistic».
- Пользователь нажимает на кнопку «SHOW STATISTICS»

Альтернативный сценарий:

1. Пользователь не нажимает на кнопку «SHOW STATISTICS».
2. Пользователь вводит вторую дату меньшую, чем первая.
3. Пользователь не вводит одну из дат.
4. Пользователь делает активным флажок «Port statistic».
5. Пользователь выполняет получение статистики по кораблям через API.

Результат:

1. На странице появляется статистика по путешествиям за указанный период в виде диаграмм, которые показывают среднюю протяжённость маршрутов, корабли, которые имеют больше всего рейсов, корабли, имеющие наибольший «пробег» за указанный период.

Сценарий – «Просмотр статистики по портам».

Действующее лицо: Пользователь

Предусловие:

1. Пользователь находится на главной странице приложения и переходит на страницу «statistics» с помощью кнопки «STATISTICS».
2. Пользователь находится на странице «statistics».

Основной сценарий:

1. Пользователь вводит промежуток в годах в два поля, которые имеют подпись «Time period».
2. Пользователь делает активным флажок «Port statistic».
3. Пользователь нажимает на кнопку «SHOW STATISTICS»

Альтернативный сценарий:

1. Пользователь не нажимает на кнопку «SHOW STATISTICS».
2. Пользователь вводит вторую дату меньшую, чем первая.
3. Пользователь не вводит одну из дат.
4. Пользователь делает активным флажок «Voyages statistic».
5. Пользователь выполняет получение статистики по портам через API.

Результат:

1. На странице появляется статистика по кораблям за указанный период в виде диаграмм, которые показывают наиболее популярные порты отправления и прибытия и наиболее популярные маршруты.

IV. МОДЕЛЬ ДАННЫХ

4.1. Схема нереляционной базы данных.

```
{
  "_id": <ObjectId>,
  "shipName": "St. Elena",
  "commander": "David Porter Jr.",
  "departure": {
    "_id": <ObjectId>,
    "name": "New York",
    "lat": 134.2144,
    "lon": 95.313
  },
  "destination": {
    "_id": <ObjectId>,
    "name": "Los Angeles",
    "lat": 89.2144,
    "lon": 33.313
  },
  "startDate": <Timestamp>,
  "endDate": <Timestamp>,
  "distance": 9356.54
}
```

Рисунок 4.1 — Схема нереляционной базы данных.

4.2. Список сущностей модели

Разработанная модель данных включает следующие коллекции: Points, SeaTrips.

4.3. Описание назначений коллекций, типов данных и сущностей.

Описание полей коллекции SeaTrips:

Таблица 1 — Описание полей коллекции SeaTrips.

| Название | Тип данных | Описание |
|-------------|------------|--|
| shipName | String | Название судна |
| commander | String | Имя командира |
| startDate | Timestamp | Дата начала путешествия |
| endDate | Timestamp | Дата окончания путешествия |
| distance | Double | Расстояние пройденное судном за время путешествия |
| departure | Object ID | Уникальный идентификатор начальной точки путешествия |
| destination | Object ID | Уникальный идентификатор конечной точки путешествия |

Описание полей коллекции Points:

Таблица 2 — Описание полей коллекции Points.

| Название | Тип данных | Описание |
|----------|------------|----------------|
| name | String | Название точки |
| lat | Double | Широта |
| lon | Double | Долгота |

4.4. Аналог модели данных для SQL СУБД.

Аналог модели данных для SQL СУБД совпадает с нереляционным.

4.5. Оценка удельного объема информации, хранимой в модели.

MongoDB ("чистый" объем):

1. SeaTrip:

`shipName` - 50B

`commander` - 50B

`startDate` - 4B

`endDate` - 4B

`distance` - 8B

2. Point:

`name` - 32B

`lat` - 8B

`lon` - 8B

Тогда чистый объем информации будет равен $N * (\text{size}(\text{SeaTrips}) + P(=2) * \text{size}(\text{Point})) = 212N$ байт, Где N - количество путешествий, P - количество точек для каждого путешествия, равное 2.

MongoDB (фактический объем):

1. SeaTrip:

`_id` - 12B

`shipName` - 50B

`commander` - 50B

`startDate` - 4B

`endDate` - 4B

`distance` - 8B

`departure` - 12B

`destination` - 12B

2. Point:

`_id` - 12B

`name` - 32B

`lat` - 8B

`lon` - 8B

Тогда фактический объем информации будет равен $N * (\text{size}(\text{SeaTrips}) + P(=2) * \text{size}(\text{Point})) = N(152 + 120) = 272N$ байт, Где N - количество путешествий, P - количество точек для каждого путешествия, равное 2.

Избыточность модели данных MongoDB: $272N/212N$

SQL ("чистый" объем):

1. SeaTrip:

`ShipName` – 50B

`Commander` – 50B

`StartDate` – 3B

`EndDate` – 3B

`Distance` – 8B

2. Point:

`Name` – 32B

`Lat` – 8B

`Lon` – 8B

Тогда чистый объем информации будет равен $N * (\text{size}(\text{SeaTrips}) + P(=2) * \text{size}(\text{Point})) = 210N$ байт, Где N - количество путешествий, P - количество точек для каждого путешествия, равное 2.

SQL (фактический объем):

1. SeaTrip:

`Id` – 4B

`ShipName` – 50B

`Commander` – 50B

`DepartureId` – 4B

`DestanationId` – 4B

`StartDate` – 3B

`EndDate` – 3B

`Distance` – 8B

2. Point:

`Id` – 4B

`Name` – 32B

`Lat` – 8B

`Lon` – 8B

Тогда чистый объем информации будет равен $N * (\text{size}(\text{SeaTrips}) + P(=2) * \text{size}(\text{Point})) = 230N$ байт, Где N - количество путешествий, P - количество точек для каждого путешествия, равное 2.

Избыточность модели данных SQL: $230N/210N$

4.6. Запросы к модели, с помощью которых реализуются сценарии использования.

Добавление путешествия (MongoDB)

`db.seaTrips.insertOne(`

```

{
  _id:1,
    shipName: "St. Elena",
    commander: "David Porter Jr.",
    startDate: new Date(41242142),
    endDate: new Date(41242142),
    distance: 9356.54,
    departure: {
      "_id": 21,
      "name": "New York",
      "lat": 134.2144,
      "lon": 95.313
    },
    destination: {
      "_id": 22,
      "name": " Los Angeles",
      "lat": 184.2144,
      "lon": 75.313
    },
  })

```

Добавление путешествия (SQL)

```

INSERT INTO Point VALUES(...)
INSERT INTO SeaTrip VALUES(...)

```

Поиск записи по id (MongoDB)

```

db.seaTrips.find({_id:1})

```

Поиск записи по id (SQL)

```

SELECT SeaTrip.ShipName, SeaTrip.Commander, Point.name as `Departure`,
Point.name as `Destination`, SeaTrip.StartDate, SeaTrip.EndDate, SeaTrip.Distance
FROM SeaTrip
LEFT JOIN Point
ON Point.Id = SeaTrip.DestinationId
LEFT JOIN Point
ON Point.Id = SeaTrip.DepartureId
WHERE SeaTrip.Id = [id]

```

Поиск путешествий в определенный временной интервал (MongoDB)

```

db.seaTrips.find(
  startDate: {
    $gte: "Sun May 30 20:40:36 +0000 2010"
  },
  endDate: {
    $lt: "Mon May 30 18:47:00 +0000 2015",
  })

```

Поиск путешествий в определенный временной интервал (SQL)

```

SELECT  SeaTrip.ShipName,  SeaTrip.Commander,  Point.name  as
`Departure`,
Point.name      as      `Destination`,      SeaTrip.StartDate,
SeaTrip.EndDate, SeaTrip.Distance FROM SeaTrip
LEFT JOIN Point
ON Point.Id = SeaTrip.DestinationId
LEFT JOIN Point
ON Point.Id = SeaTrip.DepartureId
WHERE SeaTrip.startDate > "Sun May 30 20:40:36 +0000 2010"
AND SeaTrip.startDate < "Mon May 30 18:47:00 +0000 2015"

```

V. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

5.1. Схема экранов приложения

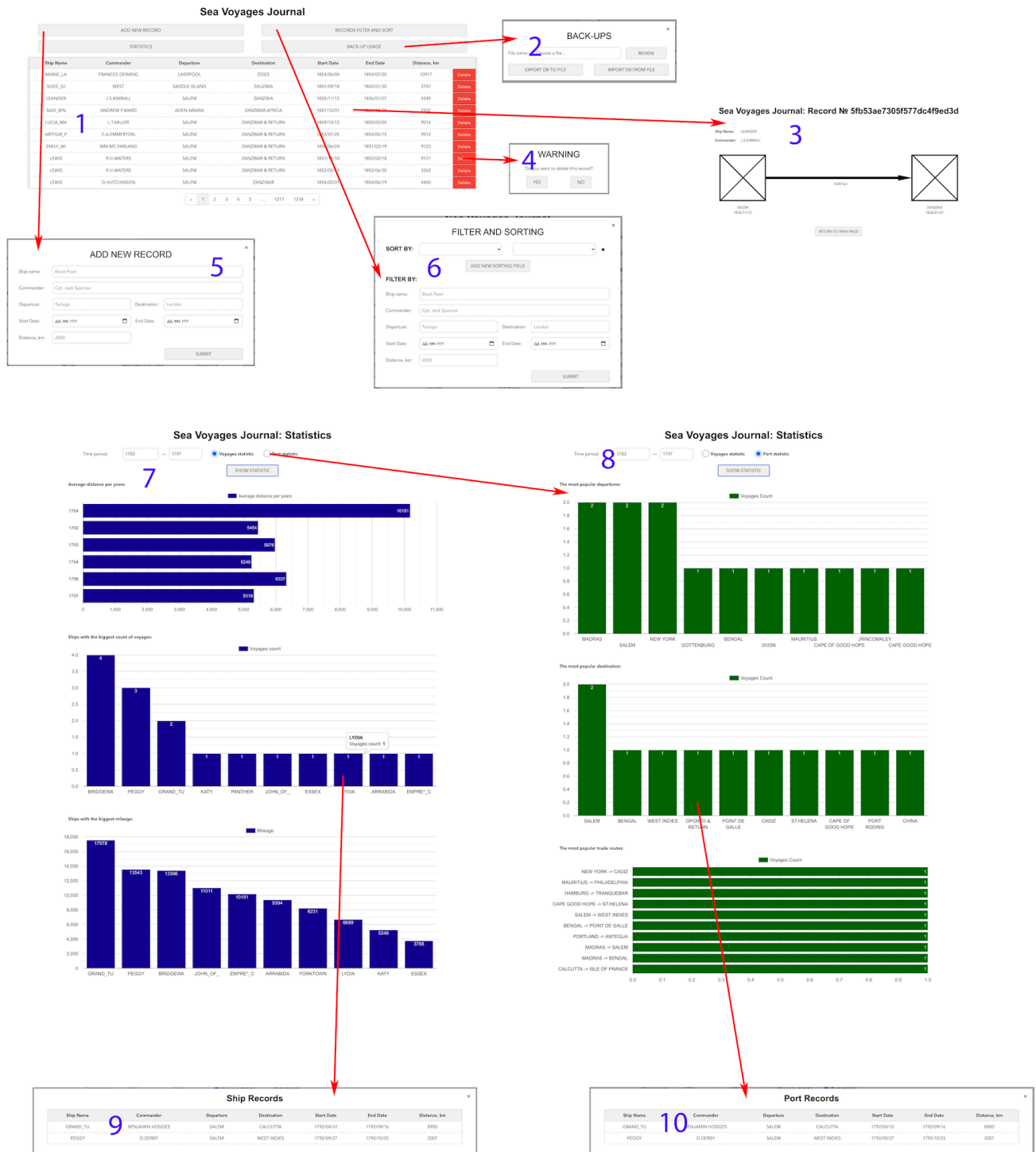


Рисунок 5.1 — Схема экранов приложения.

1. Просмотр списка морских путешествий.
2. Импорт / Экспорт БД.
3. Просмотр путешествия.
4. Удаление путешествия.
5. Добавление новой записи.
6. Добавление фильтров и параметров сортировки записей.
7. Просмотр статистики путешествий.
8. Просмотр статистики портов.
9. Подробная информация по гистограмме путешествий.
10. Подробная информация по гистограмме портов.

5.2. Использованные технологии.

Node.js, Vue.js, CSS, Express.js, Mongo.

5.3. Ссылки на приложение.

1. Github: <https://github.com/moevm/nosql2h20-sea-trips>

VI. ВЫВОДЫ

6.1. Достигнутые результаты.

Было разработано user-friendly приложение, в функциональность которого входит: страница со списком морских путешествий, содержащая записи за все время, статистика путешествий, фильтрация и сортировка путешествий, добавление путешествий, импорт и экспорт данных БД. В качестве системы управления базами данных используется MongoDB.

6.2. Недостатки и пути для улучшения полученного решения.

К недостаткам текущей реализации можно отнести грубую оценку длины маршрута, использование только формата JSON для импорта- экспорта.

6.3. Будущее развитие решения.

Дальнейшее развитие приложения предполагает увеличение числа форматом для импорта-экспорта данных, использование более точной оценки пути, увеличение числа статистик для отображения; поддержка маршрутов с промежуточными остановками.

VII. ПРИЛОЖЕНИЯ

7.1. Документация по сборке и развертыванию приложения.

Инструкция для Docker.

1. Скачать репозиторий [4].
2. Внутри папки дирекории проекта открыть терминал.
3. Выполнить команду `docker-compose up --build`.

VIII. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация Vue.js — <https://v3.vuejs.org/guide/introduction.html>.
2. Документация MongoDB — <https://docs.mongodb.com/>.
3. Документация Express.js — <https://expressjs.com/ru/starter/installing.html>
4. Github-репозиторий — <https://github.com/moevm/nosql2h20-sea-trips>